# High throughput microscopy: from raw images to discoveries

**Roy Wollman[1,*] and Nico Stuurman[2]**

[1]Department of Molecular and Cellular Biology, University of California, Davis, CA, USA
[2]The Howard Hughes Medical Institute and the Department of Cellular and Molecular Pharmacology, University of California, San Francisco, CA, USA
*Author for correspondence (e-mail: rwollman@ucdavis.edu)

*Accepted 17 September 2007*
*Journal of Cell Science 120, 3715-3722 Published by The Company of Biologists 2007*
*doi:10.1242/jcs.013623*

## Summary

**Technological advances in automated microscopy now allow rapid acquisition of many images without human intervention, images that can be used for large-scale screens. The main challenge in such screens is the conversion of the raw images into interpretable information and hence discoveries. This post-acquisition component of image-based screens requires computational steps to identify cells, choose the cells of interest, assess their phenotype, and identify statistically significant 'hits'. Designing such an analysis pipeline requires careful consideration of the necessary hardware and software components, image analysis, statistical analysis and data presentation tools. Given the increasing availability of such hardware and software, these types of experiments have come within the reach of individual labs, heralding many interesting new ways of acquiring biological knowledge.**

Key words: High-throughput microscopy (HTM), Image analysis, RNAi, Genome-wide screen

## Introduction

High-throughput microscopy (HTM) allows researchers to acquire images automatically from thousands of different treatments overnight or over several days. This technological advance makes it possible to conduct large-scale, image-based screens to discover novel genes and novel functions of familiar genes. Several such screens have been performed (Bakal et al., 2007; Berns et al., 2004; Boutros et al., 2004; Loo et al., 2007; Paran et al., 2007; Perlman et al., 2004; Rickardson et al., 2007; Tanaka et al., 2005) – for other examples see recent reviews (Carpenter and Sabatini, 2004; Echeverri and Perrimon, 2006; Mitchison, 2005; Pepperkok and Ellenberg, 2006) – and many more are yet to come. As readouts for experiments using RNA interference (RNAi) or pharmacological treatments, these image-based screens have become powerful tools for assessing cellular response to loss or perturbation of protein function.

Automation of all aspects of microscope control and the use of robotic sample preparation opened the door to rapid and large-scale screens. Two other technological advances contributed substantially. In 1965 Gordon Moore predicted that the transistor density on microchips would double every 2 years (Moore, 1998), the result being a similar doubling in computing power. Amazingly, after more than 40 years this prediction still holds true. In addition, data storage capacity has grown at an almost similar rate; at the time of writing, the storage cost of a digital image is ~$0.001, roughly two orders of magnitude lower than the cost ten years ago. Also, networking technologies now make it possible to transfer massive amounts of data between computers. Many advances in biology and bioinformatics have benefited from this exponential growth in computing ability. Further increases in storage capacity, bandwidth and processing power are expected at a staggering rate.

Advances in computer hardware were not the only contributors to the development of HTM. Image-based screens are part of a bigger genomic revolution. In particular, the discovery of RNAi (Clemens et al., 2000) and the availability of whole genome sequences allow the systematic knockdown of every gene or specific gene sets in a genome (Echeverri and Perrimon, 2006). Genome-wide RNAi screens are, of course, not the only application. The methodology needed for RNAi screens is similar to that for image-based chemical screens to identify potential drug candidates (Eggert et al., 2004). In addition to screens, automation in microscopy will have an effect on many routine microscopy applications, which will soon be augmented by automated image acquisition. For example, it allows live-cell time-lapse imaging to be performed on multiple samples for long periods (Gordon et al., 2007; Neumann et al., 2006).

The computational partner of computer-controlled microscope automation is image analysis, which often does not receive the full attention it deserves. The human brain took millions of years to evolve and is superior in its analytical capabilities to any artificial system. Since the interpretation of visual scenes and images is so natural for us, we often do not consider it to be a difficult problem. Research in computer vision started in the 1960s. At that time, pioneers in the field predicted that computers would be able to outperform the human brain by 1985 (Crevier, 1993). These predictions did not hold true and computers are still far from reaching our capacity to perceive and analyse images.

Although computers cannot replace humans in this context, they do not fatigue when confronted with massive quantities of images. Consider, for example, the number of images in a human genome-wide (~20,500 genes) RNAi screen. It is necessary to collect images from sufficient numbers of cells to reliably determine a phenotype and so one should image ~12 sites per

well. Assuming four types of staining at different wavelengths, the overall number of images acquired will be ~1,000,000. Furthermore, some siRNA methodologies require the use of multiple probes per gene, increasing this number severalfold. In ideal conditions, these can be acquired by a robotic microscope in <2 weeks. However, visual analysis of 1,000,000 images is a daunting task, often requiring far more time than the acquisition itself. Not only do computers have more stamina than most human image analysers, they also perform analysis in a much more reproducible fashion and do not suffer from bias in their analysis and classification. In some cases, computers can pick up subtle differences in phenotype that only become apparent after quantitative analysis of many cells (e.g. Huang and Murphy, 2004), which cannot be carried out by a human observer. Also, the quantitative output of computer analysis is amenable to multiple ways of classification unlike the qualitative classification a human would yield.

How then can one analyze ~1,000,000 images in a reasonable amount of time? Here we describe the requirements for such analysis and the stages that are necessary for successful large-scale image-based screens. We introduce important concepts from computer vision and computational image analysis and discuss some caveats that might hamper such screens, illustrating the different parts of the 'image-analysis pipeline' with an RNAi screen for metaphase spindle morphology in *Drosophila* S2 cells (Fig. 1) (Goshima et al., 2007).

## Setting up the analysis pipeline

To take full advantage of automated microscopy, it is important to automate as many stages as possible from acquisition to final results. A simple task that seems trivial in regular microscopy, such as copying image files from the microscope computer to another workstation for analysis, can become a major obstacle if not automated in advance. Thus, it is important to consider the entire pipeline after image acquisition. As in an industrial assembly line, at each step material is transformed and the output from one step becomes the input for the next. Raw images go through a series of analysis steps that depend on one another. Each of these steps can be seen as a building block, and in many cases, these building blocks can be reused in other screens.

Analysis pipelines can be optimized for quality and speed. Usually, the step that is slow or error-prone determines the overall quality and speed of the analysis. The analysis system is complicated and depends on many hardware and software components. It is therefore important to add reporting mechanisms that indicate when errors occur. Analysis of such errors should consider both the quality and quantity of information and accommodate as many aspects of the overall analysis process as possible. Such errors could include poor staining in one of the plates, hard drive failure, etc. These should be identified in real-time during the analysis and reported to the screen supervisor to facilitate immediate manual intervention. Such reporting (which can be as simple as automatic email notification) is an integral component of the pipeline. There is no single scheme suitable for all types of analysis. The final solution should be tailored to the specific needs of the screen.

Software components in an analysis pipeline can be written in-house, be downloaded from a free source or be purchased commercially. When writing software in-house, these should be written in such a way that they can be reused. Often, others have already written many of the routines, although it can sometimes be laborious to understand code written by others and adapt it to a specific screen. Different image-analysis platforms provide different levels of extensibility. Commercial packages usually charge for any additional module, whereas free software often has a user base that is happy to share existing code, but this might be less robust and not fully tested. It is vital to choose a set of tools that can be integrated with one another (see Table 1). Many commercial high-throughput microscopes come with a companion image-analysis package that provides some functionality for image analysis and image storage. The problem is that they are usually hard to integrate with other tools. There is no point in choosing the best tool from every category if they cannot be integrated seamlessly. Several vendors sell all-in-one systems that work 'out of the box' and include various components – a microscope, a database, the analysis software, etc. – that, if they are suitable, can bypass the need to design an analysis pipeline.

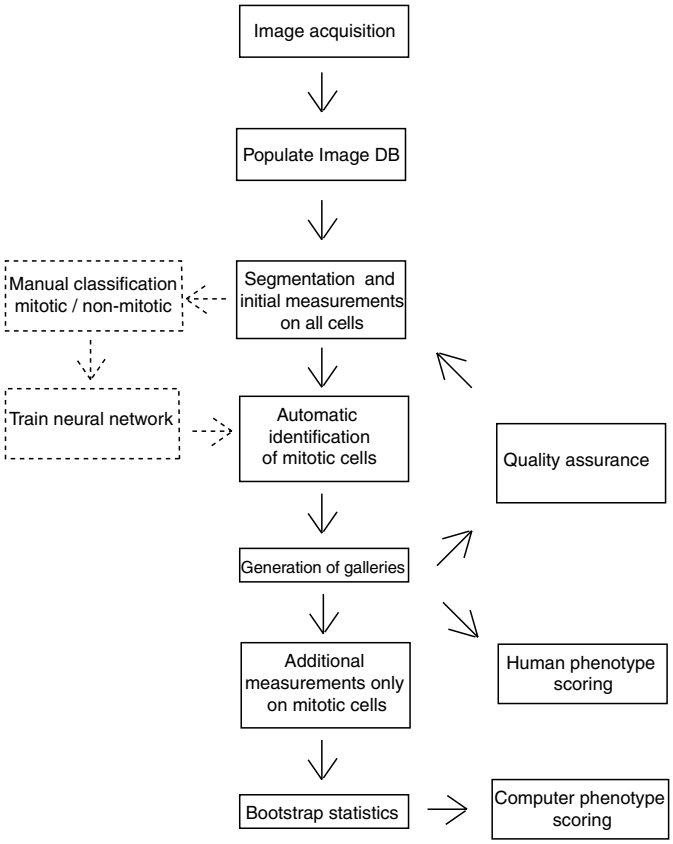


**Fig. 1.** Analysis pipeline. An example of the analysis pipeline, used in an RNAi screen of *Drosophila* spindle morphology. After acquisition, the images were streamlined to the image database. The analysis code accessed the raw images, segmented them to identify cells and classified the cells to distinguish mitotic cells from apoptotic and interphase cells. The mitotic cells were reorganized into galleries for visual analysis. The same cells from the galleries were analyzed further to identify specific mitotic phenotypes. Bootstrap statistics was used to identify wells that exhibited statistically significant extreme phenotypes.

**Table 1. Software packages and tools for use in high-throughput image analysis**

| Name | Type | Website | License, cost | Application |
|---|---|---|---|---|
| CellID | Image analysis | www.molsci.org/protocols/software/cell_id/ | Open Source, Free | Microscope-based cytometry |
| CellProfiler | Image analysis | www.cellprofiler.org/ | Open Source, Free | Image analysis package based on Matlab. Also able to execute part of an analysis pipeline |
| ImageJ | Image analysis | rsb.info.nih.gov/ij/ | Freeware | General purpose image analysis package written in Java. Not straightforward to use without a GUI |
| ImageMagick | Image conversion | www.imagemagick.org/ | Open Source, Free | Command line utilities for image file type conversion. |
| Matlab | Integrated development environment | www.mathworks.com/ | Proprietary, commercial | Development environment widely used in engineering. Image analysis capabilities and database interaction depends on toolboxes. |
| MySQL | SQL database server | www.mysql.com/ | Open Source, Free | SQL database server |
| Open Microscopy Environment | data management of microscope images | www.openmicroscopy.org/ | Open Source, Free | Tools for data management for biological light microscope images |
| phplabware | Data visualization | phplabware.sf.net/ | Open Source, Free | Web-based data management and visualization |
| PostgresQL | SQL database server | www.postgresql.org/ | Open Source, Free | SQL database server |
| R | Statistical analysis | www.r-project.org/ | Open Source, Free | Statistical analysis and graphics |
| Spotfire | Data visualization | www.spotfire.com/ | Proprietary, commercial | Visualization of data from many sources (including SQL databases) |

Below we describe a few types of 'building blocks' that are commonly used in many image analysis tasks. Although not all steps are required in all screens, they provide a useful framework and starting point for analysis-pipeline design. For additional discussion on pipeline design for image-based screen see the article by Carpenter (Carpenter, 2007).

## Segmentation – identifying the cells

The goal of the analysis pipeline is to measure specific characteristics in multiple cells for each treatment and to use appropriate statistical procedures to determine which treatments are 'hits', i.e. those that produce an extreme phenotype. The first step is to identify the cells in the images. This is accomplished by an image-analysis procedure called segmentation, which divides the image into regions such that the background is one region and each cell is a different region. For example, an image containing 100 cells is divided into 101 regions. There are many image-analysis algorithms that perform segmentation and it is important to choose the appropriate algorithm. Most segmentation processes use the following steps. First, a binary image is generated in which the foreground (cells) has a value of one and the background has a value of zero. The binary image is further processed to separate the foreground into separate 'islands' such that each cell is a specific region. Finally, all islands are labeled, such that each cell has its own identifier.

Below, we describe three of the many segmentation strategies: simple threshold, watershed and edge-detection-based segmentation. Each method exploits a different feature in the image. A more in-depth mathematical description is provided in the book by Gonzalez (Gonzalez, 2004). Segmentation methods based on minimal graph cuts (Nath et al., 2006; Sharon et al., 2006), PDEs (Xiong et al., 2006), as well as methods that combine multiple algorithms (Li et al., 2007; Wahlby et al., 2004) are beyond the scope of this Commentary.

Simple threshold, as the name suggests, identifies an appropriate intensity threshold that separates the background from the foreground. The algorithm is simple in the sense that it does not separate objects that are not naturally separated in space. Two overlapping cells would be considered as a single cell. Therefore, simple threshold is a good method to use when objects – for example cells – are far apart. In such conditions, the simple threshold method is robust, fast and requires very few predefined parameters because it determines the threshold level automatically, using characteristics of the image itself. There are several different ways to determine the threshold. Most assume that all pixels originate from two populations: a low-intensity background and a high-intensity foreground (see also Fig. 2). The threshold is determined such that it will separate pixels into these two populations appropriately by minimizing the in-class variance and maximizing the between-class variance (Otsu, 1979). The main advantage of this method is its speed, and, if the error rate is acceptable, it should be the method of choice. The method has two disadvantages. First, it is not very robust with respect to artifacts in the image. A very bright spot that is the result of a staining artifact will confuse the algorithm, making it think that the artifact is the foreground and the rest of the image is background, and thus generate meaningless results. Second, if the cells overlap or are in contact, which is often the case in culture, cells will not be separated appropriately. To limit the former problem, it is advisable to determine the threshold value using all images in the well (or even in a whole plate if the staining is uniform enough). The second problem can be overcome by sparse plating of cells. If this is not possible, other segmentation methods, such as watershed segmentation, should be explored.

The watershed segmentation algorithm starts with the same

procedure, generating a binary mask by finding an optimal threshold. However, it does not rely on the fact that objects are naturally separated; instead it assumes that the objects are round and any overlap is relatively small. The algorithm performs well when separating cells that touch each other, but it fails when confronted with large and complex cell aggregates or cells that grow on top of each other. After the generation of a binary mask, this is transformed into a 3D landscape in which the peaks are the centers of each region in the binary mask. Next, a mathematical operation equivalent to 'flooding' the landscape then defines watershed areas that separate cells with small overlaps. The watershed algorithm is more time consuming than a simple threshold and is prone to over-segmentation, separating single cells into multiple regions. An example is shown in Fig. 2M-P. Several variations of the watershed algorithm have been successfully applied to cell images (Pinidiyaarachchi and Wahlby, 2005; Wahlby and Bengtsson, 2003).

Both the simple threshold and watershed methods rely on the generation of a binary mask by identifying an optimal threshold that can separate cells from background. In some cases – for example, transient transfection in tissue culture cells – such a threshold cannot be reliably determined and other methods to create the initial binary mask are required. In such cases, it is sometimes possible to use an edge-detection algorithm. An edge-detection method relies on identifying local changes in intensity rather than a global threshold, and this is useful when there is a large variability in intensity between cells. Several edge-detection methods rely on preprocessing of the image to enhance edges. After cell edges are identified, they can be extended and connected by a series of operations termed morphological operators, which fill in the area between these edges to cover the whole cell (see Fig. 2E-L).

Because non-overlapping objects are easiest to segment, it is often advisable to identify nuclei rather than entire cells in the segmentation step. This might even be sufficient when the goal of the screen does not require cytoplasmic measurements. An alternative is to use the nuclei as 'seeds' and grow objects around those seeds out to the boundaries of the cells. In the spindle screen mentioned above (Goshima et al., 2007), the cells were relatively flat and non-overlapping. This allowed straightforward simple threshold segmentation of the DAPI-stained nuclei. A required distance of 16 μm (100 pixels) between cells avoided over-segmentation. Other examples of screens where this simple approach was successful have been described (Moffat et al., 2006; Pelkmans et al., 2005).

## Classification – dividing cells into populations of interest

When analysing phenotypes we frequently want to observe only a subset of the cell population – for example, when only a specific subset is of interest or when a subset of the population should be avoided (e.g. apoptotic cells). This requires a labeling step (classification), which computers can

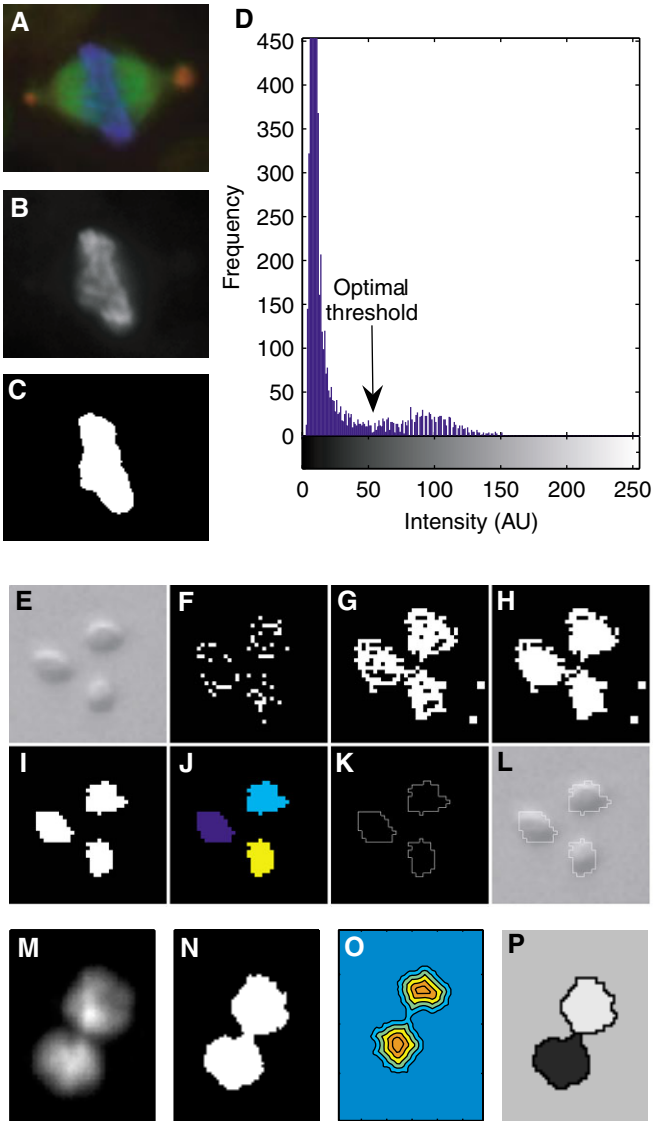


**Fig. 2.** Segmentation. (A-D) Segmentation using the simple threshold method. The DAPI channel (B) from the cell shown in A is used for segmentation. The histogram (D) of intensity values for the DAPI channel shows bimodality. A background population of pixels averages an intensity of 20 and a foreground population of pixels averages an intensity of 100. The optimal threshold (an intensity of 50) is the intensity that separates those two population while maximizing the between-population variance and minimizing the within-population variance. (C) Binary image created by using the optimal threshold. (E-L) Segmentation using edge detection. Yeast cells imaged in DIC that are unsuitable for threshold-based segmentation can be segmented by the edge-detection algorithm. (E) Raw image of the yeast cells. Results of an edge detection (F) operation are then improved by morphological dilation (G) and a morphological operation that fills the 'holes' in the image. Owing to the initial dilation, the binary masks are too large and are reduced by morphological erosion (I). The results of the segmentation are three unconnected yeast cells that can be labeled individually. Images K and L show the final edge and an overlay of the edge on the original cells. (M-P) Segmentation using watershed operation. Nuclei of *Drosophila* S2 cells stained with DAPI (M) are in close proximity and a simple threshold (N) does not segment them properly. A distance transformation assigns each pixel inside the cell a value equal to its distance from the edge. This creates an image that can be interpreted as 3D topographical landscape containing two 'hills' (O). The watershed transformation separates the hill by a mathematical operation that is equivalent to 'raising the water level' until all hills are separated; the result is seen in P.

be trained to perform. One can do this by manually labeling a small sample of cells from all categories and training the computer to identify them and optimize some parameters to minimize misclassification. The result is a classifier: a mathematical function that, given an input of cells, will output a set of labels associated with each cell. This type of procedure is termed supervised learning and is distinct from clustering or unsupervised learning, in which the computer divides the input cells into categories without prior training (Duda, 2001). Supervised learning is a large and very active field of study, including artificial neural networks, support vector machines, statistical Bayesian classifiers, etc. Most of the approaches follow similar concepts and have similar caveats. From a user's perspective, there is very little difference because in all cases the computer needs to be trained and the algorithms act in a similar way. The main differences concern the performance of the classifier, which can be affected by many parameters other than the underlying algorithm. One can substantially improve its performance by changing the image-analysis steps prior to classification and by changing the feature set used in the classifier. In most cases, such improvements are more substantial than any achieved by a change in the learning algorithm. Finally, note that classification can also be used to identify subcellular regions (see Conrad et al., 2004; Glory and Murphy, 2007).

To classify cells, the computer uses a set of cell measurements. The specific classification problem dictates the choice of cell measurements, such as nuclear area, fluorescence intensity of multiple channels in different cell regions, and morphological characteristics such as the circumference-to-area ratio of the cell. It is tempting to generate very large feature matrices that are all encompassing and contain many columns and let the computer sort out what it needs on the basis of the training set. In practice, this often yields poor results. This is the curse of dimensionality: the larger the number of features (columns in the feature matrix), the bigger the training set required. One should therefore choose the minimal set of features necessary to distinguish between different cell categories. There are formal algorithms that can help decide which features should be included in the classification, but these algorithms should be used as guidelines only, and a manual judgment can be very helpful. To choose the features, it is helpful to look at projections of the entire dataset onto 2D or 3D scatter plots and to choose those features that show separation in these projections (see Fig. 3).

After the successful choice of input features, classifier training can begin. The size of the training set depends on the difficulty of the classification, but a few hundred manually labeled cells should be enough for simple classification tasks. Given the feature matrix and the manually classified set of labels, the optimization process usually takes only a few minutes of computer time. When the optimization is over, the classifier is ready to classify unseen cells. It is important to avoid the problem of over-fitting. Sometimes, when the complexity of the classifier and the feature set is unnecessarily large, the classifier will perform very well on the training set but will not be able to generalize its knowledge to an unseen sample. One should therefore evaluate the classifier's performance on an unseen dataset before it is deployed to the large-scale screen.

To evaluate the classifier, additional manual labeling should be compared with the classifier's labels of the same dataset. The classifier is then evaluated by the percentage of misclassification in this unseen dataset. There are two types of error: false positives (cells that are wrongly classified in the target class) and false negatives (cells that are missing from the target class). There is a trade-off between these two errors; a more permissive classifier will miss fewer hits (generate fewer false negatives) but will generate more false positives. To evaluate a classifier and gauge the trade-off between these two error types, receiver operating characteristic (ROC) curves are often used (Lasko et al., 2005).

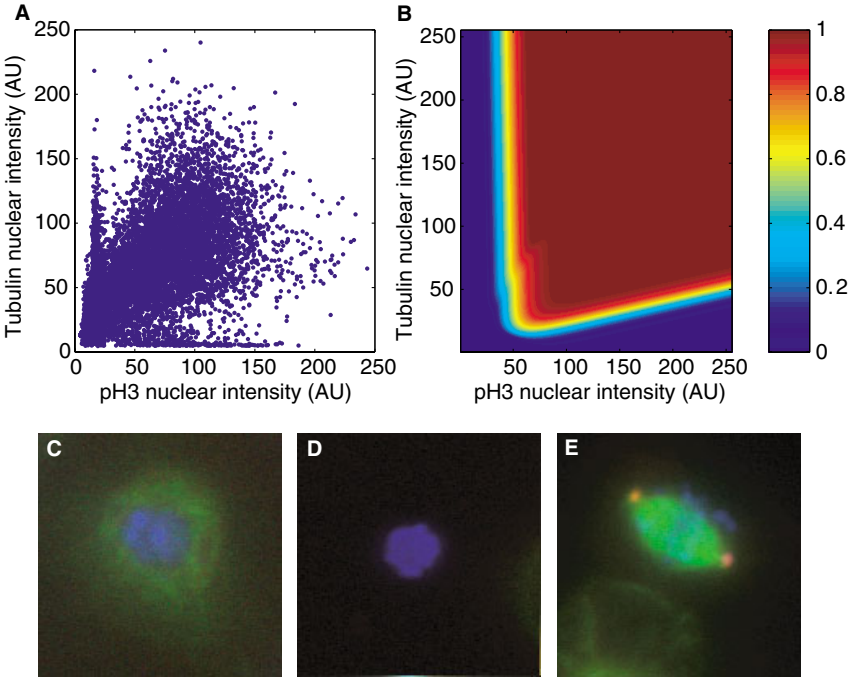


**Fig. 3.** Cell classification. Cells were labeled with an antibody binding phosphorylated histone 3 (PH3, marking mitotic cells), an antibody binding tubulin and with DAPI (labeling DNA). (A-E) Scatter plot of a population of cells (A) showing three distinct populations: interphase cells pH3 intensity<25 and tubulin nuclear intensity 0–150 (see C); apoptotic cells, pH3 intensity 0-150 and tubulin intensity<20 (see D); and mitotic cells, high pH3 intensity and tubulin staining (see E). Cells can be classified as mitotic and non-mitotic using a simple artificial neural network. The result of the training is shown as a heat map (B) of the probability that a cell is mitotic. Mitotic (and apoptotic) cells can be discriminated from interphase cells solely on the basis of pH3 intensity; however, since both mitotic and apoptotic cells stain well with this antibody, an additional feature (tubulin staining) is needed to discriminate between the two. Notice that, although in this simple example a neural network is not strictly necessary and good classification could be achieved using two thresholds, training of artificial classifiers scales better with dimensionality of the inputs and does not require manual tinkering to determine classification parameters.

### Measuring the phenotypes

At this stage, the analysis has identified the cells in the image and chosen the cells on which to perform measurements. The next natural step is then to perform measurements: what is the phenotype after a specific treatment? This is a very screen-specific question, one that should be addressed in the early stages of assay development. There are two approaches. The simplest and often superior one is to have a specific hypothesis in advance and measure only the characteristics that are relevant for this hypothesis. This is the case when screening for changes in protein localization or a specific phenotype, such as change in spindle length. In the RNAi screen of *Drosophila* spindle morphology (Goshima et al., 2007), for example, spindle length could be determined by identifying γ-tubulin-stained centrosomes in all spindles that had only two centrosomes in their Voroni 'region of influence', and measuring the distance between them. In this case, computer analysis was superior to manual inspection and allowed identification of known factors (Eb1) and novel genes (*ssp4*) that were otherwise missed by the human observer.

At the other end of the spectrum, there are cases where no specific hypothesis is tested: are there any 'morphology' changes after the treatment, for example? To answer such questions, it is advisable to perform multiple measurements that detect a wide spectrum of potential phenotypes. Typical features are cell area, overall intensity or texture (variance of intensity level). It is important to choose characteristics that are independent of each other to glean the most information about the measured cell. Characteristics that correlate well – for example, cell area and cell diameter – will not add new information and only interfere with further analysis. In cases where multiple measurements are performed, use computational tools to simplify the dataset before proceeding to the final stage of 'hit' identification. In many cases the same biological phenomenon can be manifested in several measurements. For example, cell area, actin intensity and the number of focal adhesions are all aspects of cell spreading. It is possible to transform these three measurements into a single characteristic that will represent spreading. This can be achieved using principal components analysis (PCA), a statistical procedure that reduces dimensionality. PCA first creates a new coordinate system (components) consisting of a linear combination of the existing features. The new components are sorted according to their contribution to the overall variance. One can then choose a few components that represent most of the variability in the feature set. It is sometimes informative to check what feature combination is identified by PCA. For example, if in a mitotic screen one of the major components is a linear combination of the number of chromosome spots, a shape score of the chromosome area and the DAPI intensity, and if all other features contribute very little information to this component, then this component could be related to aneuploidy.

Two examples of screens that scored overall morphology and used dimensionality reduction techniques to summarize overall morphology are described in Tanaka et al. and Bakal et al. (Tanaka et al., 2005; Bakal et al., 2007). Tanaka et al. tested changes in morphology as a result of treatment with 107 small molecule compounds. To test changes in morphology, 38 different measurements where performed for each cell. The measurements included characteristics such as cell area, nuclear area, average tubulin intensity, etc. PCA was used to project these 38 measurements into a 3D space for ease of visualization. In this, the 107 compounds partitioned into four clusters. This allowed the identification of inhibitors of cellular components not targeted by known protein kinase inhibitors. In their screen, Bakal et al. calculated 145 features for each cell. As dimensionality-reduction techniques, they used an alternative to PCA that allows non-linear projections. They first identified seven classes of morphology based on known phenotypes that are result of perturbation of key signaling molecules (e.g. Rho and Rac). Then, using the 145 feature matrix, they trained a set of artificial neural networks to recognize these. The result is a matrix with seven columns; each represents the similarity of the cell to the phenotypic category. In essence, they created a transformation that projects cells from a 145-dimensional space of quantitative measurements into a 7-dimensional space of similarity to predefined phenotypic categories. This transformation was the basis of further hierarchical cluster analysis that allowed identification of local signaling networks that regulate cell protrusion, adhesion and tension.

### Identifying the hits

Even in control wells, natural variability alone means that we expect to have some extreme values, especially when the number of wells is very high. How can we identify treatments that are indeed 'hits' – i.e. those that cause a phenotype more extreme than expected by chance?

The traditional statistical approach to answering such questions is to estimate the probability of getting such an extreme phenotype in the absence of treatment (the *P*-value). Large screens present several challenges to rigorous statistical analysis. There are many types of measurement performed that do not follow a normal distribution, preventing the use of common statistical tools such as *t*-tests and analysis of variance (ANOVA). In large screens, the number of phenotypes upon which to perform statistical tests can be very high (>100,000), thus increasing the likelihood that these will yield 'hits', regardless of any real effect. One must therefore use statistical procedures to correct for this (Storey and Tibshirani, 2003). Furthermore, in many screens, there are no formal controls (such as no RNAi, scrambled RNAi or RNAi directed against a gene that is not thought to yield any phenotype) but instead all the wells that are imaged undergo some sort of treatment. And finally, in many cases there will be phenotypic variability between plates that could be the result of small variations in the fixation and staining protocols. These challenges require application of appropriate statistical procedures to help prevent false positives and make the overall results more reliable.

To estimate a *P*-value for a specific extreme phenotype, we need to consider what the null hypothesis is. What does a population of normal wild-type (untreated) cells look like and what is the probability that the measured phenotype originated from such a population? There are several ways to estimate this, mainly differing in their assumptions. In the simplest case a plate has a few control wells containing wild-type cells, and the phenotype distribution approximates a normal distribution. The straightforward student *t*-test would therefore suffice. However, in many cases, the screen design does not include wells containing untreated cells. This is reasonable because most treatments are unlikely to produce a phenotype and,

therefore, can be used to estimate the 'normal' population. Resampling can be used to estimate the wild-type distribution. Take, for example, a well containing 37 cells in which the average cell area is 100 $\mu m^2$. One can estimate the probability of getting such a value by chance by sampling many sets of 37 cells from the plate and seeing what percentage of these samples has an area smaller or bigger than 100 $\mu m^2$. Choose at random 1,000,000 sets of 37 cells from the plate and use these 1,000,000 virtual wells as a measure of the wild-type distribution. If the average cell area for the first well is smaller than those for 997,567 of these virtual wells, then we can estimate the *P*-value to be (1,000,000-997,567)÷1,000,000=0.002433. This procedure is computationally intensive; however, it has the advantage of not assuming any prior knowledge about the phenotypic distribution, and it uses cells from the same plate as an internal control. It still needs to be determined whether a *P*-value of 0.002433 is significant.

False positives are perhaps the biggest challenge in identifying true hits. The main cause is the large number of tests performed in a screen. In a human genome screen using 25,000 siRNAs and four phenotypic measurements, >100,000 statistical tests will be performed. If the traditional *P*-value cutoff of 0.05 is used, we expect to get 5000 hits by chance alone. The most straightforward remedy is to lower the cutoff – but to what extent? The most stringent method, the Bonfferoni correction maintains the chance of a single false positive at 0.05 by dividing 0.05 by the number of tests. In the above example, this would lower the cutoff to 0.0000005. This method deals with the problem of false positives very well. However, by decreasing the number of false positives, we lose statistical power, and the number of false negatives will increase. A less stringent method, the false discovery rate (FDR) (Storey and Tibshirani, 2003), uses a less stringent correction. FDR decreases the cutoff *P*-value such that no more than 5% of the hits are expected to be false positives. It therefore misses fewer true hits. In general, it is advisable to use several cutoff values and divide the hits into different levels of confidence. These potential hits can then be verified in a secondary screen.

## Infrastructure – where to store all the data

Before starting a large-scale screen, it is vital to set up the appropriate computational infrastructure. As mentioned previously, HTM generates vast quantities of data (~1,000,000 images at 2.5 MB per image amounts to 2.5 TB). This imposes a huge burden on the computing infrastructure, both hardware and software. The hardware should have sufficient capacity in terms of storage, processing power and network bandwidth. It is advisable to have, in addition to the acquisition control unit, three other units: a storage unit, a processing unit and a backup unit. Storage will probably take place on arrays of hard drives (RAIDs) that have some redundancy built in. Nevertheless, backing up on another system, preferably off-site and/or on another medium, is crucial to safeguard acquired images and analysis results. In the RNAi screen of *Drosophila* spindle morphology (Goshima et al., 2007), a 2TB RAID stored the raw images. The raw images were backed up on a second RAID. They were then compressed and archived. Since analysis comprises multiple small tasks that can be performed in parallel, one can speed up analysis by using multiple processing units and/or processing units composed of multiple CPUs and/or cores. To avoid bottlenecks, communication between the various units should be as fast as possible (1 GB per second is currently very affordable). The spindle morphology screen used two computers, each with two CPUs, for image analysis and to perform statistical tests. Communication between computers use Gigabit Ethernet.

The software should provide three types of service: storage management, analysis and final presentation. There are multiple types of data that require storage and retrieval: raw images, acquisition meta-data, treatment types, intermediate analysis steps, extracted measurements and final statistics. A central storage solution that can deal with all data types is advisable and eliminates the need for compatibility layers between several databases. The analysis platform should be able to communicate with the central database, perform the segmentation, measurements and classifications, and perform statistical tests. There are many possible frameworks and software packages that can facilitate these types of analysis (see Table 1). At the end of the screen, a large amount of data will be generated and, potentially, numerous people will want to access it. Many database solutions allow the presentation of a subset of the results to different groups; such solutions are advisable and simplify the final data presentation. The spindle morphology screen used a central database server running PostgresQL, analysis was performed using Matlab, and final presentation of the results and processed images used phplabware, a web-based database system (Table 1). A more in depth review on the infrastructure needed for large-scale image-based screens can be found elsewhere (Vaisberg et al., 2006).

## Concluding remarks

HTM and technological advances in computer hardware and software made large-scale image-based screens a reality. Today, genome-wide image-based screens are simple enough to perform in a single lab. As automation of image acquisition becomes ubiquitous, the importance of post-acquisition analysis is becoming clearer. Post-acquisition analysis is still the bottleneck for large-scale screens but future development of more powerful computers and algorithms will help mitigate this further. Still, it is important to remember that computer vision is far behind in some of its capabilities compared with human vision. The key to success in HTM large-scale screens is designing smart and powerful assays that take the capabilities and limitations of automated image analysis into account. When designing these analysis pipelines, there is no need to reinvent the wheel and rewrite from scratch all the computational routines needed for a screen. There are many existing tools, both commercial and open-source, that provide excellent functionality in all the steps mentioned above (see Table 1). We foresee that the increasing availability of libraries of compounds and RNAi reagents, automated microscopes and computer infrastructure for high-throughput analysis will make image-based genome-size screens available to individual labs as just another tool to address a wide-variety of interesting questions in creative, novel ways.

This work was funded by grants from the University of California Systemwide Biotechnology Research & Education Program GREAT Training Grant (2006-03) to R.W. We thank Ron Vale for helpful suggestions and Lisa Petrie for help in editing the manuscript.

## References


**Bakal, C., Aach, J., Church, G. and Perrimon, N.** (2007). Quantitative morphological signatures define local signaling networks regulating cell morphology. *Science* **316**, 1753-1756.

**Berns, K., Hijmans, E. M., Mullenders, J., Brummelkamp, T. R., Velds, A., Heimerikx, M., Kerkhoven, R. M., Madiredjo, M., Nijkamp, W., Weigelt, B. et al.** (2004). A large-scale RNAi screen in human cells identifies new components of the p53 pathway. *Nature* **428**, 431-437.

**Boutros, M., Kiger, A. A., Armknecht, S., Kerr, K., Hild, M., Koch, B., Haas, S. A., Paro, R. and Perrimon, N.** (2004). Genome-wide RNAi analysis of growth and viability in Drosophila cells. *Science* **303**, 832-835.

**Carpenter, A. E.** (2007). Image-based chemical screening. *Nat. Chem. Biol.* **3**, 461-465.

**Carpenter, A. E. and Sabatini, D. M.** (2004). Systematic genome-wide screens of gene function. *Nat. Rev. Genet.* **5**, 11-22.

**Clemens, J. C., Worby, C. A., Simonson-Leff, N., Muda, M., Maehama, T., Hemmings, B. A. and Dixon, J. E.** (2000). Use of double-stranded RNA interference in Drosophila cell lines to dissect signal transduction pathways. *Proc. Natl. Acad. Sci. USA* **97**, 6499-6503.

**Conrad, C., Erfle, H., Warnat, P., Daigle, N., Lorch, T., Ellenberg, J., Pepperkok, R. and Eils, R.** (2004). Automatic identification of subcellular phenotypes on human cell arrays. *Genome Res.* **14**, 1130-1136.

**Crevier, D.** (1993). *AI: The Tumultuous History of the Search for Artificial Intelligence*. New York: Basic Books.

**Duda, R. O.** (2001). *Pattern Classification*. New York: Wiley.

**Echeverri, C. J. and Perrimon, N.** (2006). High-throughput RNAi screening in cultured cells: a user's guide. *Nat. Rev. Genet.* **7**, 373-384.

**Eggert, U. S., Kiger, A. A., Richter, C., Perlman, Z. E., Perrimon, N., Mitchison, T. J. and Field, C. M.** (2004). Parallel chemical genetic and genome-wide RNAi screens identify cytokinesis inhibitors and targets. *PLoS Biol.* **2**, e379.

**Glory, E. and Murphy, R. F.** (2007). Automated subcellular location determination and high-throughput microscopy. *Dev. Cell* **12**, 7-16.

**Gonzalez, R. C.** (2004). *Digital Image Processing using MATLAB*. Upper Saddle River, NJ: Pearson Prentice Hall.

**Gordon, A., Colman-Lerner, A., Chin, T. E., Benjamin, K. R., Yu, R. C. and Brent, R.** (2007). Single-cell quantification of molecules and rates using open-source microscope-based cytometry. *Nat. Methods* **4**, 175-181.

**Goshima, G., Wollman, R., Goodwin, S. S., Zhang, N., Scholey, J. M., Vale, R. D. and Stuurman, N.** (2007). Genes required for mitotic spindle assembly in Drosophila S2 cells. *Science* **316**, 417-421.

**Huang, K. and Murphy, R. F.** (2004). Boosting accuracy of automated classification of fluorescence microscope images for location proteomics. *Bmc Bioinformatics* **5**, 78.

**Lasko, T. A., Bhagwat, J. G., Zou, K. H. and Ohno-Machado, L.** (2005). The use of receiver operating characteristic curves in biomedical informatics. *J. Biomed. Inform.* **38**, 404-415.

**Li, F., Zhou, X., Ma, J. and Wong, S. T.** (2007). An automated feedback system with the hybrid model of scoring and classification for solving over-segmentation problems in RNAi high content screening. *J. Microsc.* **226**, 121-132.

**Loo, L. H., Wu, L. F. and Altschuler, S. J.** (2007). Image-based multivariate profiling of drug responses from single cells. *Nat. Methods* **4**, 445-453.

**Mitchison, T. J.** (2005). Small-molecule screening and profiling by using automated microscopy. *Chembiochem* **6**, 33-39.

**Moffat, J., Grueneberg, D. A., Yang, X., Kim, S. Y., Kloepfer, A. M., Hinkle, G., Piqani, B., Eisenhaure, T. M., Luo, B., Grenier, J. K. et al.** (2006). A lentiviral RNAi library for human and mouse genes applied to an arrayed viral high-content screen. *Cell* **124**, 1283-1298.

**Moore, G. E.** (1998). Cramming more components onto integrated circuits [reprinted from Electronics, pp. 114-117, April 19, 1965]. *Proc. Ieee* **86**, 82-85.

**Nath, S. K., Palaniappan, K. and Bunyak, F.** (2006). Cell segmentation using coupled level sets and graph-vertex coloring. In *Medical Image Computing and Computer-Assisted Intervention – Miccai 2006, 9th International Conference, Copenhagen, Denmark, October 1-6, 2006, Proceedings, Pt 1* (*Lecture Notes in Computer Science, Vol. 4190*) (ed. R. Larsen, M. Nielsen and J. Sporring), pp. 101-108. Berlin: Springer.

**Neumann, B., Held, M., Liebel, U., Erfle, H., Rogers, P., Pepperkok, R. and Ellenberg, J.** (2006). High-throughput RNAi screening by time-lapse imaging of live human cells. *Nat. Methods* **3**, 385-390.

**Otsu, N.** (1979). Threshold selection method from gray-level histograms. *Ieee Trans. Syst. Man Cybern.* **9**, 62-66.

**Paran, Y., Ilan, M., Kashman, Y., Goldstein, S., Liron, Y., Geiger, B. and Kam, Z.** (2007). High-throughput screening of cellular features using high-resolution light-microscopy; application for profiling drug effects on cell adhesion. *J. Struct. Biol.* **158**, 233-243.

**Pelkmans, L., Fava, E., Grabner, H., Hannus, M., Habermann, B., Krausz, E. and Zerial, M.** (2005). Genome-wide analysis of human kinases in clathrin- and caveolae/raft-mediated endocytosis. *Nature* **436**, 78-86.

**Pepperkok, R. and Ellenberg, J.** (2006). Innovation – high-throughput fluorescence microscopy for systems biology. *Nat. Rev. Mol. Cell Biol.* **7**, 690-696.

**Perlman, Z. E., Slack, M. D., Feng, Y., Mitchison, T. J., Wu, L. F. and Altschuler, S. J.** (2004). Multidimensional drug profiling by automated microscopy. *Science* **306**, 1194-1198.

**Pinidiyaarachchi, A. and Wahlby, C.** (2005). Seeded watersheds for combined segmentation and tracking of cells. In *ICIAP 2005, International Conference on Image Analysis and Processing, Cagliari, Italy, September 6-8, 2005* (*Lecture Notes in Computer Science, Vol. 3617*), pp. 336-343. Berlin, Heidelberg: Springer.

**Rickardson, L., Wickstrom, M., Larsson, R. and Lovborg, H.** (2007). Image-based screening for the identification of novel proteasome inhibitors. *J. Biomol. Screen.* **12**, 203-210.

**Sharon, E., Galun, M., Sharon, D., Basri, R. and Brandt, A.** (2006). Hierarchy and adaptivity in segmenting visual scenes. *Nature* **442**, 810-813.

**Storey, J. D. and Tibshirani, R.** (2003). Statistical significance for genomewide studies. *Proc. Natl. Acad. Sci. USA* **100**, 9440-9445.

**Tanaka, M., Bateman, R., Rauh, D., Vaisberg, E., Ramachandani, S., Zhang, C., Hansen, K. C., Burlingame, A. L., Trautman, J. K., Shokat, K. M. et al.** (2005). An unbiased cell morphology-based screen for new, biologically active small molecules. *PLoS Biol.* **3**, 764-776.

**Vaisberg, E. A., Lenzi, D., Hansen, R. L., Keon, B. H. and Finer, J. T.** (2006). An infrastructure for high-throughput microscopy: instrumentation, informatics, and integration. *Meth. Enzymol.* **414**, 484-512.

**Wahlby, C. and Bengtsson, E.** (2003). Segmentation of cell nuclei in tissue by combining seeded watersheds with gradient information. In *Proceedings of SCIA-03, Scandinavian Conference on Image Analysis, Gothenberg, Sweden, July 2003* (*Lecture Notes in Computer Science, Vol. 2749*), pp. 408-414. Berlin, Heidelberg: Springer.

**Wahlby, C., Sintorn, I. M., Erlandsson, F., Borgefors, G. and Bengtsson, E.** (2004). Combining intensity, edge and shape information for 2D and 3D segmentation of cell nuclei in tissue sections. *J. Microsc.* **215**, 67-76.

**Xiong, G. L., Zhou, X. B. and Ji, L.** (2006). Automated segmentation of Drosophila RNAi fluorescence cellular images using deformable models. *Ieee Trans. Circuits Syst. I Regul. Pap.* **53**, 2415-2424.